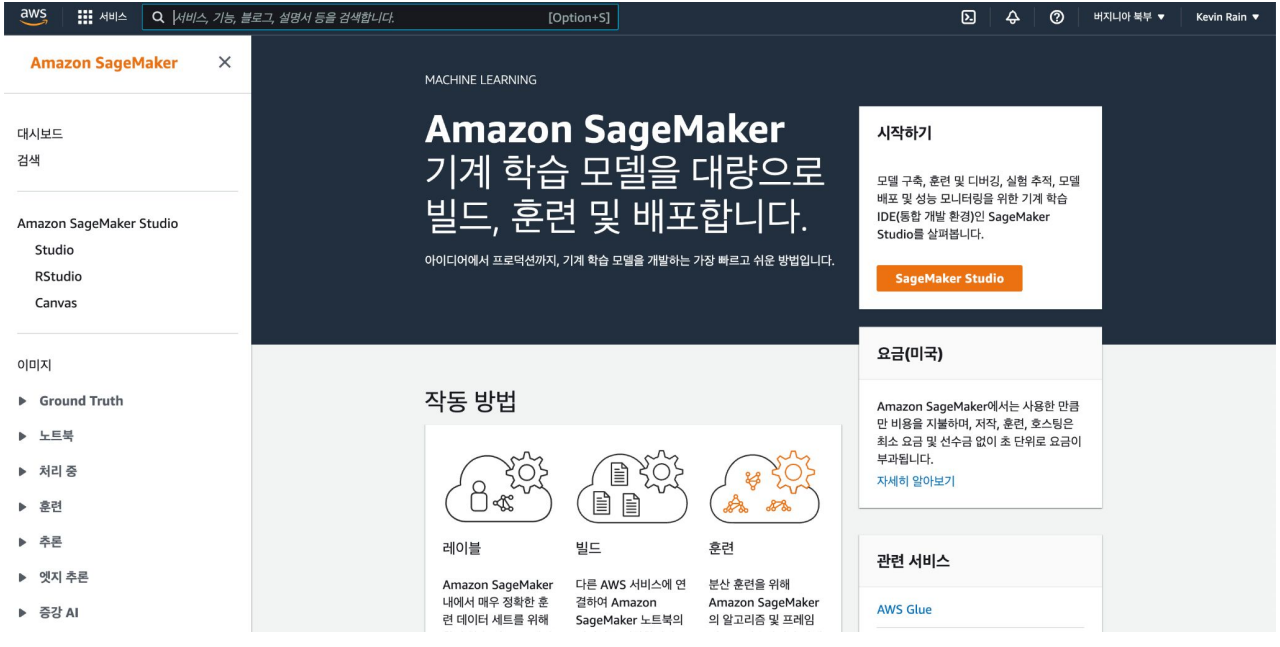
Amazon SageMaker로 ML 모델 구축, 훈련, 배포

기계학습 모델을 구축, 학습 및 배포

1. Amazon SageMaker 콘솔 열기

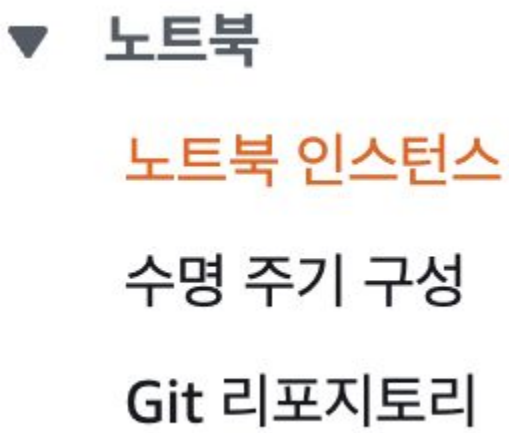
[콘솔 홈]

검색창: Amazon SageMaker



2. Amazon SageMaker 노트북 인스턴스 생성

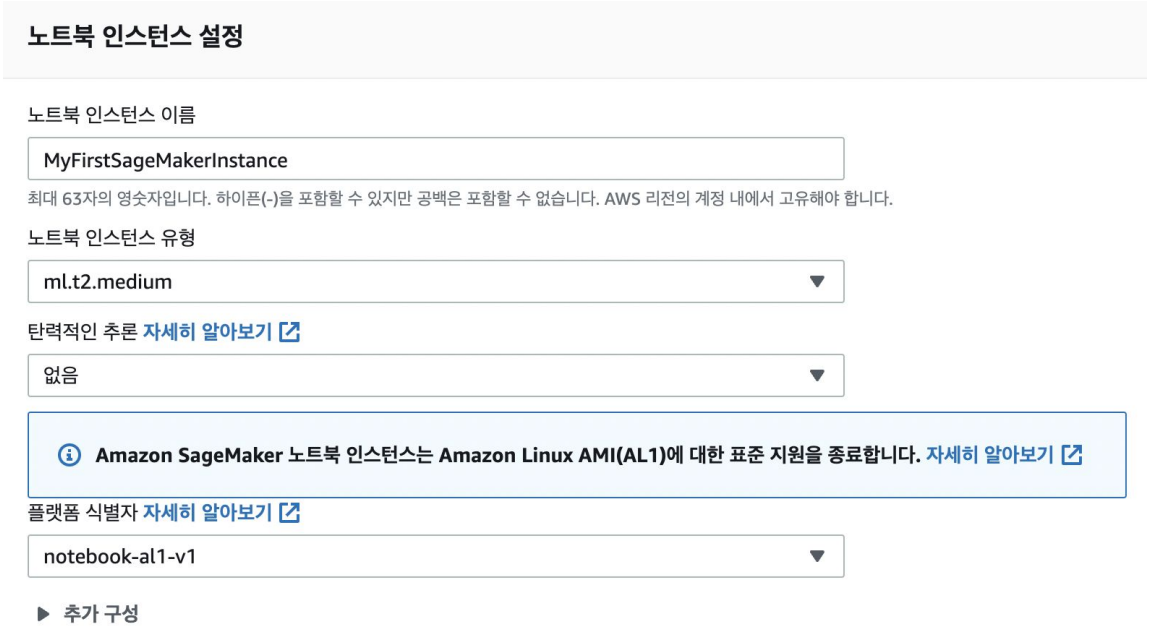
대시보드에서 노트북 인스턴스 선택



2. Amazon SageMaker 노트북 인스턴스 생성

노트북 인스턴스 생성

노트북 인스턴스 유형: ml.t2.medium



2. Amazon SageMaker 노트북 인스턴스 생성

노트북 인스턴스가 데이터에 액세스하고 안전하게 Amazon S3에 데이터를 업로드하도록 허용하려면 IAM 역할을 지정해야한다.

IAM 역할 > 새 역할 생성 > 모든 S3 버

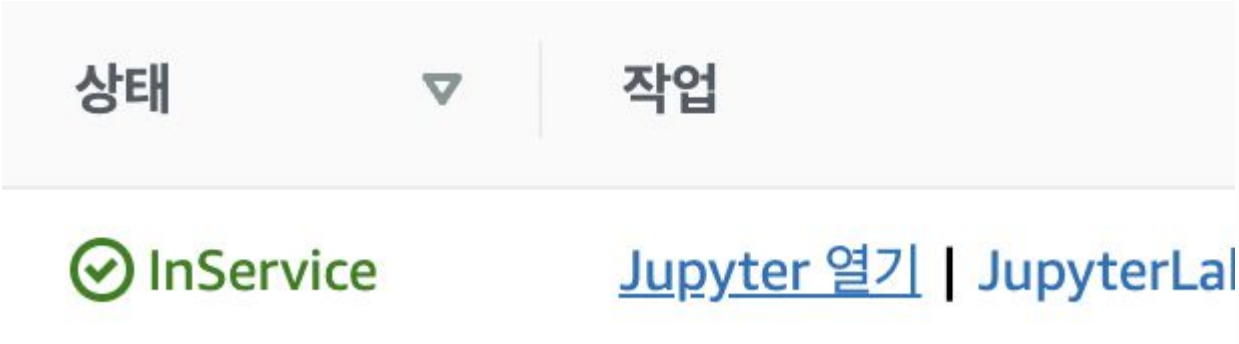
노트북 인스턴스 생성

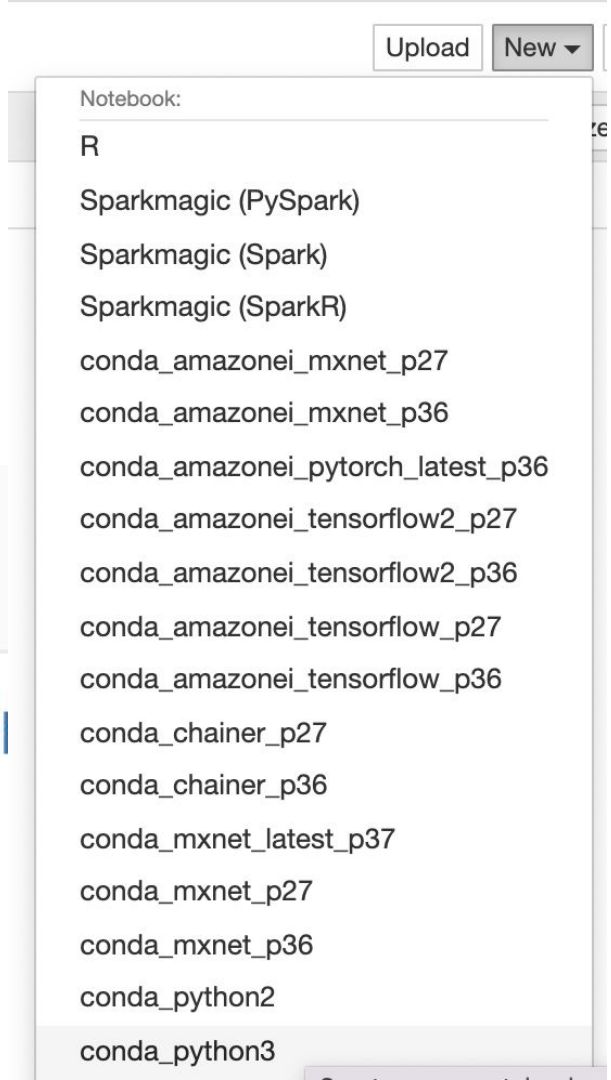
>> Pending에서 InService로 바뀜

3. 데이터 준비

InService 전환 후, 인스턴스 선택하여

작업 > Jupyter 열기





3. 데이터 준비

데이터 저장할 S3 버킷 생성

In [3]:
```
bucket_name = 'first-s3-bucket-kevin'
s3 = boto3.resource('s3')
try:
    if  my_region == 'us-east-1':
      s3.create_bucket(Bucket=bucket_name)
    else:
      s3.create_bucket(Bucket=bucket_name, CreateBucketConfiguration={ 'LocationConstraint': my_region })
    print('S3 bucket created successfully')
except Exception as e:
    print('S3 error: ',e)
```

```
S3 bucket created successfully
```

3. 데이터 준비

데이터를 Amazon SageMaker 인스턴스에 다운로드 & dataframe에 load

In [4]:
```
try:
  urllib.request.urlretrieve ("https://d1.awsstatic.com/tmt/build-train-deploy-machine-learning-model-sagemaker/bank_cl
  print('Success: downloaded bank_clean.csv.')
except Exception as e:
  print('Data load error: ',e)

try:
  model_data = pd.read_csv('./bank_clean.csv',index_col=0)
  print('Success: Data loaded into dataframe.')
except Exception as e:
    print('Data load error: ',e)
```

```
Success: downloaded bank_clean.csv.
Success: Data loaded into dataframe.
```

3. 데이터 준비

데이터 shuffle & split (7:3)

In [5]:
```
train_data, test_data = np.split(model_data.sample(frac=1, random_state=1729), [int(0.7 * len(model_data))])
print(train_data.shape, test_data.shape)
```

```
(28831, 61) (12357, 61)
```

4. 데이터에서 모델 훈련

앞서 정의한 모델 사용하기 위해 훈련 데이터 형식 수정 & S3 버킷에서 데이터 로드

AttributeError Fix

In [9]:
```
pd.concat([train_data['y_yes'], train_data.drop(['y_no', 'y_yes'], axis=1)], axis=1).to_csv('train.csv', index=False, h
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix, 'train/train.csv')).upload_file('train.c
s3_input_train = sagemaker.TrainingInput(s3_data='s3://{}/{}/train'.format(bucket_name, prefix), content_type='csv')
```

4. 데이터에서 모델 훈련

Amazon SageMaker 세션 설정 & 모델 인스턴스 생성 & 모델 하이퍼파라미터 정의

Reamed Estimator Parameters

In [12]:
```
sess = sagemaker.Session()
xgb = sagemaker.estimator.Estimator(containers[my_region],role, instance_count=1, instance_type='ml.m4.xlarge',output_
xgb.set_hyperparameters(max_depth=5,eta=0.2,gamma=4,min_child_weight=6,subsample=0.8,silent=0,objective='binary:logist
```

4. 데이터에서 모델 훈련

ml.m4.xlarge 인스턴스에서 모델 훈련

```
In [15]: xgb.fit({'train': s3_input_train})
```

```
2022-01-25 01:52:39 Starting - Starting the training job...
2022-01-25 01:53:05 Starting - Launching requested ML instancesProfilerReport-1643075559: InProgress
.....
2022-01-25 01:54:08 Starting - Preparing the instances for training.....
2022-01-25 01:56:06 Downloading - Downloading input data...
2022-01-25 01:56:26 Training - Downloading the training image...
2022-01-25 01:57:06 Training - Training image download completed. Training in progress.Arguments: train
[2022-01-25:01:57:02:INFO] Running standalone xgboost training.
[2022-01-25:01:57:02:INFO] Path /opt/ml/input/data/validation does not exist!
[2022-01-25:01:57:02:INFO] File size need to be processed in the node: 3.38mb. Available memory size in the node: 834
6.91mb
[2022-01-25:01:57:02:INFO] Determined delimiter of CSV input is ','
[01:57:02] S3DistributionType set as FullyReplicated
[01:57:02] 28831x59 matrix with 1701029 entries loaded from /opt/ml/input/data/train?format=csv&label_column=0&delimi
ter=,
[01:57:02] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 30 extra nodes, 14 pruned nodes, max_depth=5
[0]#011train-error:0.100482
[01:57:02] src/tree/updater_prune.cc:74: tree pruning end, 1 roots, 30 extra nodes, 14 pruned nodes, max_depth=5
[1]#011train-error:0.099858
```

5. 모델 배포

모델을 배포하고 액세스할 수 있는 엔드포인트를 서버에 생성

In [16]:
```
xgb_predictor = xgb.deploy(initial_instance_count=1,instance_type='ml.m4.xlarge')
```

```
----------!
```

5. 모델 배포

테스트 데이터 이용해서 모델 예측 수행

AttributeError: can’t set attribute

The csv_serializer has been renamed

In [18]:
```
xgb_predictor.__dict__.keys()
```

Out[18]:
```
dict_keys(['endpoint_name', 'sagemaker_session', 'serializer', 'deserializer', '_endpoint_config_name', '_model_name
s', '_context'])
```

In [21]:
```
from sagemaker.serializers import CSVSerializer
```

In [22]:
```
test_data_array = test_data.drop(['y_no', 'y_yes'], axis=1).values #load the data into an array
# xgb_predictor.content_type = 'text/csv' # set the data type for an inference
xgb_predictor.serializer = CSVSerializer() # set the serializer type
predictions = xgb_predictor.predict(test_data_array).decode('utf-8') # predict!
predictions_array = np.fromstring(predictions[1:], sep=',') # and turn the prediction into an array
print(predictions_array.shape)
```

```
(12357,)
```

6. 모델 성능 평가

Confusion Matrix 테이블에서 실제값 vs 예측값 비교

In [24]:
```
cm = pd.crosstab(index=test_data['y_yes'], columns=np.round(predictions_array), rownames=['Observed'], colnames=['Predi
tn = cm.iloc[0,0]; fn = cm.iloc[1,0]; tp = cm.iloc[1,1]; fp = cm.iloc[0,1]; p = (tp+tn)/(tp+tn+fp+fn)*100
print("\n{0:<20}{1:<4.1f}%\n".format("Overall Classification Rate: ", p))
print("{0:<15}{1:<15}{2:>8}".format("Predicted", "No Purchase", "Purchase"))
print("Observed")
print("{0:<15}{1:<2.0f}% ({2:<}){3:>6.0f}% ({4:<})".format("No Purchase", tn/(tn+fn)*100,tn, fp/(tp+fp)*100, fp))
print("{0:<16}{1:<1.0f}% ({2:<}){3:>7.0f}% ({4:<}) \n".format("Purchase", fn/(tn+fn)*100,fn, tp/(tp+fp)*100, tp))
```

```
Overall Classification Rate: 89.5%

Predicted       No Purchase     Purchase
Observed
No Purchase     90% (10769)     37% (167)
Purchase         10% (1133)      63% (288)
```

7. 리소스 종료

S3 버킷에서 Amazon SageMaker 엔드포인트와 객체를 삭제

The endpoint attribute has been renamed

In [26]:
```
sagemaker.Session().delete_endpoint(xgb_predictor.endpoint_name)
bucket_to_delete = boto3.resource('s3').Bucket(bucket_name)
bucket_to_delete.objects.all().delete()
```